12 avril 2016

Automatisation de tests d'accessibilité chez Orange, on s'y met et vous ?

Vincent Aniort (@sanvin) et Fabien Legris

orange™

Qui sont ces gens là ?

- Service de support aux projets de développement
- Outils, recommandations et méthodes
- Vincent, expert accessibilité numérique
- Fabien, architecte logiciel

Contenu

1. Rappels
2. Que voulions nous faire ?
3. Qu'avons-nous fait ?
4. Résultats et conclusion

Rappels,

vite fait !

Accessibilité
définition de l'accessibilité

c'est un service web utilisable par tous

- personnes valides
- seniors
- personnes en situation de handicap (temporaire ou permanent)

et dans tous les contextes

- avec tous types de matériel : navigateurs, PC, MAC, tél. mobiles, tablettes
- dans un contexte dégradé : mauvaise luminosité, touchpad en mobilité, etc.
- avec des logiciels spécifiques de compensation du handicap

Accessibilité
référentiels d'accessibilité

Recommandations internationales édictées par le W3C/WAI

W3c organisme de normalisation à but non lucratif, chargé de promouvoir la compatibilité des technologies du World Wide Web

WAI (Web Accessibility Initiative) groupe de travail du W3c

- WCAG (Web Content Accessibility Guidelines)
  - WCAG 1 (1999)
  - WCAG 2 (2008), devenu une norme ISO/IEC 40500:2012
- RGAA v3 méthode d'application des WCAG 2.0 en France
- WAI-ARIA 1.0 (Accessible Rich Internet Application) langage permettant des ajouts sémantiques entre autre au HTML et assurant la navigation clavier

Accessibilité
référentiels d'accessibilité

3 niveaux d'accessibilité :

- **niveau A** : environ 1/3 des critères

  fournir un équivalent textuel à tout contenu non textuel, une navigation clavier…

- **niveau AA** : environ 2/3 des critères

  fournir une structure sémantique pour le contenu, un contraste suffisant, des titres de rubrique et des étiquettes…

- **niveau AAA** : TOUS les critères

  fournir la signification des abréviations, une audiodescription et un sous-titrage pour les vidéos…

Techniques WCAG 2.0

Non normatif, permet de guider les développeur dans la mise en place des recommandations d'accessibilité

(général, HTML, CSS , SMIL, Client-side Scripting, Server-side Scripting, Flash, PDF, Silverlight, WAI-ARIA)

Forge interne d'intégration continue

Git : gestion de versions décentralisée

Node.js :

- une plateforme basée sur le moteur Javascript V8 (Chrome)
- plateforme de construction logicielle (web)
- modules installables avec le gestionnaire de paquets *NPM (Node Package Manager*)

NodeJS est ici utilisé pour la partie outillage du développeur

Grunt : gestionnaire de tâches : pour créer et exécuter des tâches de build

Jenkins : application d'intégration et de delivery en continu

Tout cet environnement est Open Source et libre, adapté à l'automatisation des tâches, très utilisé dans la communauté des développeurs, évolue rapidement et fréquemment.

Que voulions-nous faire ?

Notre outillage …

Accessibilité et automatisation tests

30% des critères d'accessibilité automatisables, plus de 50% des erreurs

Il faut tester le code interprété par le navigateur,
et pas seulement le code statique (Rich Internet Applications)

Convaincre et diffuser l'accessibilité :

- ❑ automatiser au maximum
- ❑ intégration dans une chaine de tests cohérente prêt à l'emploi
- ❑ identification et donc correction des erreurs d'accessibilité
- ❑ le site/application assure les basiques de l'accessibilité de manière indolore
- ❑ et donc le projet fait un pas dans la mise en place de l'accessibilité

Orange Boosted with Bootstrap framework interne open source

Choix de se baser sur Bootstrap : mobile first, RWD (responsive web design), grosse activité de la communauté, largement répandu (en interne), prise en main aisée, facilement extensible ou customisable...

Orange Boosted with Bootstrap :

- Open Source (licence MIT, déposé sur GitHub boosted.orange.com)
- de conception centrée utilisateur (amélioration continue)
- suivi des évolutions de Bootstrap
- des composants spécifiques et chartés Orange, ergonomiques, accessibles , interopérables et adaptatifs
- enrichir et homogénéiser l'expérience utilisateur

Constat suite aux retours projets : même avec des composants unitaires d'interface propres, on peut arriver à avoir une page "sale" !

-> fournir l'outillage complet de tests de qualité de code front pour s'assurer de l'amélioration progressive de la qualité de code et de l'accessibilité, par la même occasion

Qualité de code (front), performance (front), accessibilité ***tests automatisés***

Utiliser des outils pour mettre en place une chaine complète et gérer l'amélioration progressive

Qualité de code :

- htmllint, validation du HTML
- csslint, validation des CSS
- eslint, validation JS

Tests spécifiques :

- bootlint, validation de la syntaxe spécifique BootStrap
- lesslint ou scsslint, validation des LESS/SASS

Optimisation des performances :

- concaténation, minification, uglification...

Mais jusque-là rien de très avant-gardiste, maintenant intégrons dans cette chaine l'accessibilité !

Accessibilité et automatisation
outils étudiés

Nombreux outils identifiés :

grunt-accessibility, Pa11y, a11y, achecker, arialinter, Quail, accesslint, axe, Tota11y, AATT

Parmi ces outils, on cherche un outil de test :

- ❑ complet au niveau des critères WCAG 2.0 automatisable (avec de l'ARIA)
- ❑ léger et facilement intégrable dans un environnement Node/Grunt
- ❑ Open Source et gratuit (démarche volontaire interne)

Qu'avons-nous fait ?

Notre framework d’accessibilité

Accessibilité et automatisation choix de Axe

Axe de Deque, car :

- ❑ open source (politique interne),
- ❑ complet en termes de nombre de test,
- ❑ modulaire (paramétrable),
- ❑ intégrable (fonctionnant dans l'environnement des développeurs d'Orange et avec de nombreux frameworks de tests),
- ❑ contextuel (le périmètre des tests dépend du contexte à tester),
- ❑ extensible (on peut y ajouter des règles facilement).

Le manifeste d'Axe assure que les règles des tests :

- ❑ ne génèrent aucun faux positif,
- ❑ sont légères et s'exécutent rapidement,
- ❑ sont interopérables dans les navigateurs modernes,
- ❑ sont elles-mêmes testables automatiquement (tests unitaires).

Les tests d'accessibilité machinerie

Axe possède un des référentiels de règles
le plus complet de tests automatiques d'accessibilité (44 tests)

quelques exemple de règles :

- alternatives aux contenu non textuels (3 règles : images, area map, object, input type=image)
- contraste fond/texte
- conformité ARIA (7 règles)
- structure sémantique des titres (2 règles)

Les tests d'accessibilité machinerie

Tests en local – avant commit

- ❑ Injection du plugin de test aXe dans les pages de l'application
- ❑ Résultats disponibles en console
- ❑ Mise en avant des erreurs dans la page possible
- ❑ Possibilité de le restituer dans la page

La machinerie Axe en local html dans la page

Il y a moyen de jouer sur le formatage pour la sortie du JSON de axe

Un exemple de sortie dans la page html :

La machinerie Axe sur le serveur d'intégration

Tests automatique lors des build (Jenkins)

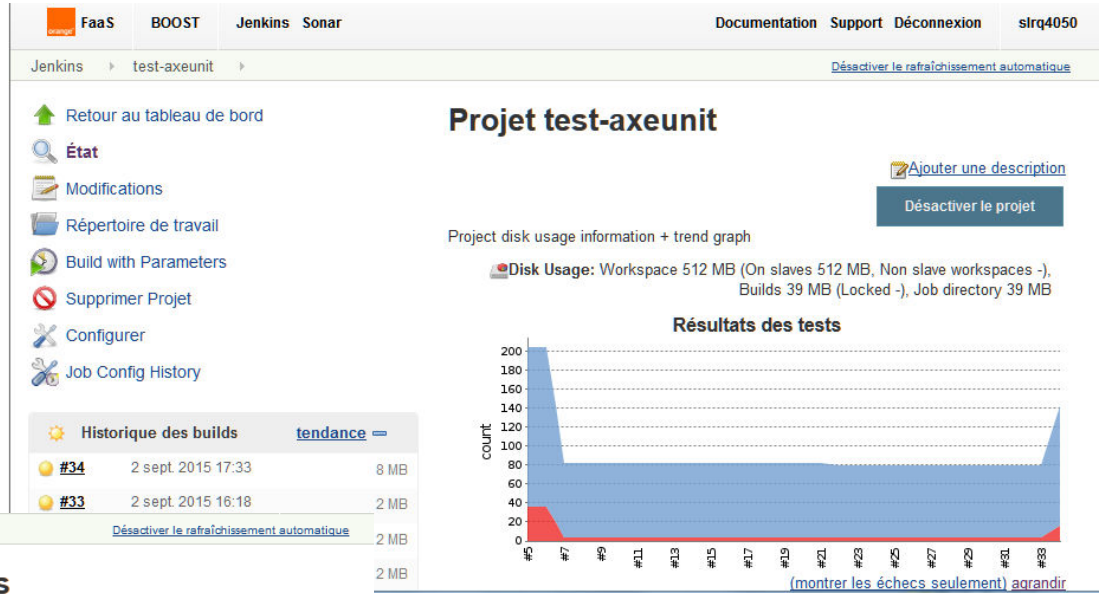
- ❑ Appel de l'application déployée lors du build au travers de CasperJS
- ❑ Enregistrement du code source audité
- ❑ Injection du moteur aXe à la volée, lancement des tests d'accessibilité
- ❑ Mise en forme de la sortie au format Xunit

La machinerie Axe lors des builds

CasperJS + Axe dans Jenkins

sortie Xunit intégration dans Jenkins :

- tableau de bord
- détails des erreurs
- statistiques par commit



Et enfin …

résultats et conclusion

Résultats

Une chaine de tests de qualité de code et accessibilité prête à l'emploi :

- ❑ faciliter l'intégration front en proposant un set de composants tout prêt,
- ❑ sensibiliser et former les équipes à la qualité de code front,
- ❑ s'assurer que les projets partent sur de bonnes bases avec leur code front, limiter la dette technique front et améliorer la maintenabilité,

Pour au final

- ❑ faire entrer l'accessibilité au plus tôt et l'intégrer au processus de développement, en faire une brique comme une autre de la qualité web.
- ❑ à partir de cette expérience dans un environnement node/grunt/jenkins, mettre en place une chaine de qualité dans d'autres environnements (robot framework, sonar...)

Conclusion
tests d'accessibilité automatisés

Vérifier la conformité face à des recommandations d'accessibilité
quelque soit la technologie employée
afin de sensibiliser et mettre en place la démarche

- ❑ nombreuses solutions
- ❑ rien de prêt à l'emploi
- ❑ adaptables à vos habitudes de travail
- ❑ à intégrer dans la chaine de tests

… et après, étape 1

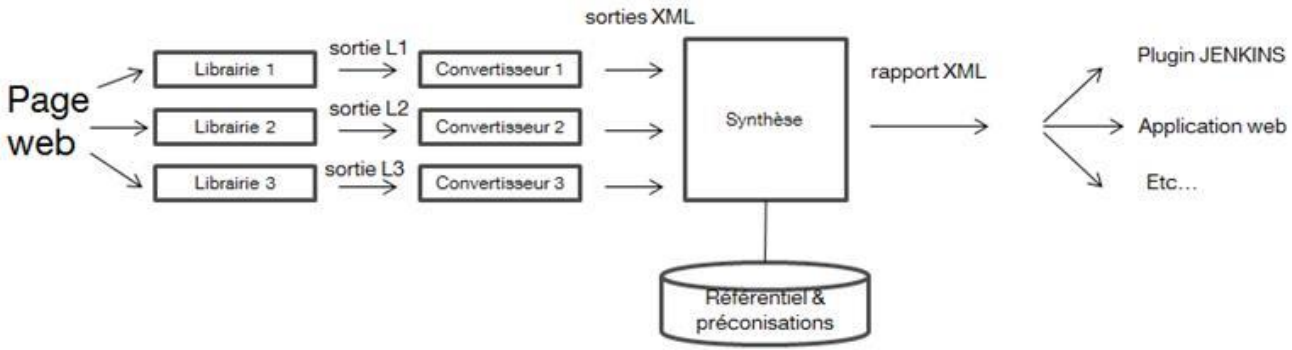
Dans un futur proche :

- ❑ industrialiser le framework (installation plus facile, support)
- ❑ améliorer l'interface utilisateur (fonctionnel, ergonomie, utilisabilité)

… et après, étape 2

Dans un futur plus lointain :

- ❑ rajouter d'autres outils existants (HTML_Code_Sniffer, Tota11y)
- ❑ étendre le socle de règles (développement à façon)



Merci

Des questions … ?

orange™

CFTL Comité Français des Tests Logiciels

Journée Française des Tests Logiciels 2016

ISTQB